Documentation Complète du Projet AuthGateAPI

Table des Matières

1. Introduction
    a) Présentation du Projet
    b) Objectifs
    c) Public Cible
2. Architecture et Technologies
    a) Stack Technique
    b) Structure des Packages
    c) Diagramme d'Architecture
3. Fonctionnalités Principales
   Authentification et Autorisation
    a) Gestion des Utilisateurs et Rôles
    b) Sécurité Avancée
    c) Audit et Historique
4. Acteurs et Rôles
    a) Administrateurs
    b) Utilisateurs Standards
    c) Développeurs Intégrant l'API
5. Guide d'Utilisation
    a) Installation et Configuration
    b) Endpoints Principaux (Swagger/OpenAPI)
    c) Exemples de Requêtes
6. Sécurité et Bonnes Pratiques
    a) Politiques de Sécurité
    b) Gestion des Tokens
    c) Prévention des Attaques
7. Contribuer au Projet (Open Source)
    a) Comment Contribuer
    b) Normes de Code
    c) Roadmap
8. FAQ & Dépannage
    a) Problèmes Courants et Solutions
    b) Support

1. Introduction

a) Présentation du Projet

AuthGateAPI est une solution complète d'authentification et d'autorisation sécurisée pour applications modernes. Elle offre :

✅ JWT (JSON Web Tokens) avec gestion des refresh tokens

✅ 2FA (Two-Factor Authentication) par email

✅ Gestion fine des permissions (RBAC)

✅ Protection contre les attaques (CSRF, Bruteforce, etc.)

✅ Audit des modifications (Hibernate Envers)

b) Objectifs

🔷 Fournir une API modulaire et sécurisée pour l'authentification

🔷 Simplifier l'intégration de la sécurité dans les applications

🔷 Offrir une gestion centralisée des utilisateurs et permissions

c) Public Cible

- Développeurs Backend cherchant une solution d'authentification prête à l'emploi
- Administrateurs Système devant gérer les accès
- Équipes DevOps souhaitant une API scalable

2. Architecture et Technologies

a) Stack Technique:

Backend : Java 17 + Spring Boot 3

Sécurité : Spring Security, JWT, Hibernate Envers (Audit)

Base de Données : PostgreSQL / MySQL (JPA)

Cache: Spring Caching (ConcurrentMap)

Documentation: Swagger/OpenAPI

Tests : JUnit, Mockito (à compléter)

b) Structure des Packages

📁 authgate-api/

├── 📁 audit/ → Logs des modifications

├── 📁 configuration/ → Configuration Spring

├── 📁 controllers/ → Endpoints API

├── 📁 exceptions/ → Gestion des erreurs

├── 📁 models/ → Entités JPA

├── 📁 repositories/ → Couche d'accès aux données

├── 📁 security/ → JWT, CSRF, Rate Limiting

├── 📁 services/ → Logique métier

└── 📁 scheduling/ → Tâches planifiées

3. Fonctionnalités Principales

🔐 Authentification et Autorisation

- ✓ Inscription avec vérification par email
- ✓ Connexion avec JWT + Refresh Token
- ✓ 2FA (Code envoyé par email)
- ✓ Rôles et Permissions (RBAC)

👥 Gestion des Utilisateurs

- ✓ CRUD Utilisateurs
- ✓ Verrouillage/Déverrouillage des comptes
- ✓ Gestion des sessions (en ligne/hors ligne)

🛡️ Sécurité Avancée

- ✓ Rate Limiting (10 requêtes/min)
- ✓ CSRF Protection (pour /logout)
- ✓ Blacklist des Tokens révoqués

📊 Audit et Historique

- ✓ Hibernate Envers pour tracer les modifications
- ✓ Endpoint /admin/users/revisions pour voir l'historique

4. Acteurs et Rôles

a) Administrateur : Gère les utilisateurs, rôles, permissions et surveille l'activité.
b) Utilisateur Standard : Se connecte, modifie son profil, réinitialise son mot de passe.
c) Développeur : Intègre AuthGate dans une application via les endpoints REST.

5. Guide d'Utilisation

a) Installation et configuration

git clone https://github.com/votrecompte/authgate-api.git

```
spring:
  datasource:
    url: jdbc: postgresql://localhost:5432/authgate
    username: admin
    password: securepassword
jwt:
  secret: votre-secret-jwt
```

b) Endpoint principaux

POST /api/auth/login : Connexion (renvoie JWT)

POST /api/auth/register : Inscription

GET /api/users/me : Récupérer son profil

POST /admin/users/lock : Verrouiller un compte (Admin seulement)

c) Exemples de requêtes

(Lien vers la documentation Swagger : http://localhost:8080/swagger-ui.html)

6. Sécurité et Bonnes Pratiques

🔒 Politiques de Sécurité

Mots de passe : Stockés en BCrypt (force 12)

Tokens JWT : Expiration configurable (15 min par défaut)

Refresh Tokens : Stockés en base (SHA-256)

🛡️ Prévention des Attaques

Rate Limiting sur /login (10 tentatives/min)

CSRF activé pour les actions sensibles (logout)

Blacklist des tokens révoqués

7. Contribuer au Projet (Open Source)

a) Comment Contribuer ?

- ✓ Forker le projet
- ✓ Créer une branche :
- ✓ git checkout-b feature/nouvelle-fonctionnalite

b) Roadmap

- ✓ Ajouter OAuth2 (Google, GitHub)
- ✓ Support Multi-langue (i18n)
- ✓ Intégration avec Keycloak

8. FAQ & Dépannage

a) Problèmes Courants

❌ "Erreur 401 Invalid Token" → Vérifiez l'expiration du JWT

❌ "Rate Limit Exceeded" → Attendez 1 minute ou augmentez la limite

Support

✉️ Contact : carlogbossou93@gmail.com

🐞 Issues GitHub github.com/davilla1993/authgate-api/issues